JP6290204

Publication Title:

METHOD OF ITERATIVE SOLUTION WITH PREPROCESSING

Abstract:

PURPOSE:To prevent deterioration in convergence due to a decrease in vector length, an overhead due to the utilization of a list vector, and paralleling as to the iterative solution of large-scale rule space simultaneous linear equations.

CONSTITUTION:Memory access at the time of forward/backward substitution in inverse approximating operation for the iterative solution is facilitated by rearranging an array to improve the operation efficiency of a pipeline processor for vector arithmetic. Further, the forward/backward substitution is performed twice for some part in an area to prevent the deterioration in convergence due to paralleling and the inverse approximating operation which is converged at nearly the same frequency of iteration with a conventional method which is small in parallelism.

---

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平6－290204

(43)公開日　平成６年(1994)10月18日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/328		9194－5L		
15/347	Ａ	9194－5L		

審査請求　有　　請求項の数２　ＯＬ　（全 11 頁）

(21)出願番号　特願平5－78404

(22)出願日　平成５年(1993)４月６日

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号

(72)発明者　鷲尾　巧
東京都港区芝五丁目７番１号　日本電気株式会社内

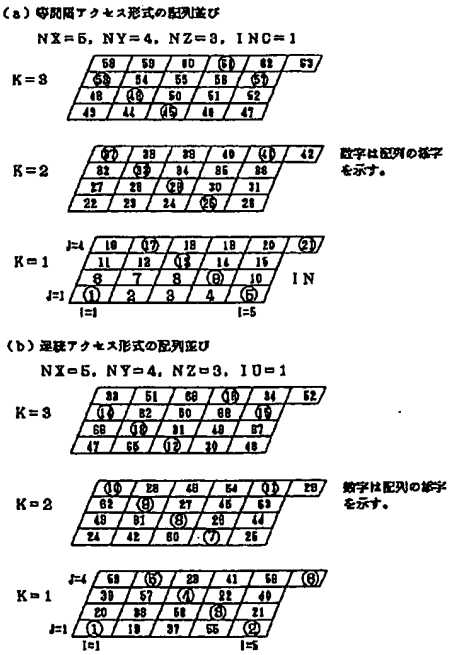
(74)代理人　弁理士　京本　直樹　（外２名）

(54)【発明の名称】　前処理付き反復解法方式

(57)【要約】

【目的】　ベクトル計算機を利用した大規模規則スパース連立一次方程式の反復求解において、ベクトル長が短くなること、リストベクトルの利用によるオーバーヘッド、並列化による収束性の悪化を防ぐ。

【構成】　配列の並べ換えにより、反復解法の逆近似操作における前進後退代入時のメモリアクセスを簡単化し、ベクトル演算でのパイプラインプロセッサの稼働効率を高める。また、前進後退代入を領域内の一部で２度行うことにより、並列化による収束性の悪化を防ぎ、並列性の少なかった従来法と同程度の反復回数で収束する逆近似操作を行う。



1

【特許請求の範囲】

【請求項１】パイプラインプロセッサと、複数のメモリバンクよりなりインターリーブ方式を採用した主記憶装置とからなるベクトル計算機システムでの前処理付き反復解法方式において、

偏微分方程式のｎ点差分近似により得られる係数行列の配列データＡ１，Ａ２，…，Ａｎ、右辺の配列データＢ、反復の初期値の配列データＵ、格子点サイズＮＸ，ＮＹ，ＮＺ、多重度ＫＤを入力データとし、前記格子点サイズを参照してＸＹ平面ごとの増分を算出し、各ＸＹ平面の配列データの末尾に前記ＸＹ平面ごとの増分を追加することにより前記係数行列の配列データ、右辺の配列データ、反復の初期値の配列データを等間隔アクセス形式に並べ換えるデータ並べ換え手段と、

前記並べ換え手段で並べ換えられた配列データと多重度ＫＤを参照して、主記憶装置への等間隔アクセスのベクトル演算で前記係数行列の近似行列分解を行い近似行列分解の配列データを生成する多数係数行列分解手段と、

前記多重度、前記係数行列分解手段で得た配列データ、前記並べ換え手段で得た係数行列の配列データを参照して、主記憶装置への等間隔アクセスのベクトル演算で前進後退代入を行う多重前進後退代入手段とからなる前処理付き反復解法方式。

【請求項２】パイプラインプロセッサと、複数のメモリバンクよりなりインターリーブ方式を採用した主記憶装置とからなるベクトル計算機システムでの前処理付き反復解法方式において、

偏微分方程式のｎ点差分近似により得られる係数行列の配列データＡ１，Ａ２，…，Ａｎ、右辺の配列データＢ、反復の初期値の配列データＵ、格子点サイズＮＸ，ＮＹ，ＮＺ、多重度ＫＤを入力データとし、前記格子点サイズを参照して、前記係数行列の配列データ、右辺の配列データ、反復の初期値の配列データを請求項１記載の等間隔アクセス形式への並べ換え手段により等間隔アクセス形式に並べ換えたのち、等間隔にならんだデータを連続に並べ換える連続アクセス形式へのデータ並べ換え手段と、

前記並べ換え手段で並べ換えられた配列データと多重度ＫＤを参照して、主記憶装置への連続アクセスのベクトル演算で前記係数行列の近似行列分解を行い近似行列分解の配列データを生成する多重係数行列分解手段と、

前記多重度、前記係数行列分解手段で得た配列データ、前記並べ換え手段で得た係数行列の配列データを参照して、主記憶装置への連続アクセスのベクトル演算で前進後退代入を行う多重前進後退代入手段とからなる前処理付き反復解法方式。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、偏微分方程式の差分近似より生じる連立一次方程式の前処理付き反復解法に関し、特に複数のメモリバンクからなりインターリーブ方式をとる主記憶装置とパイプラインプロセッサからなるベクトル計算機システムでの前処理付き反復解法方式に関する。

【０００２】

【従来の技術】科学技術計算においては、偏微分方程式を解かねばならない場合がよく生じる。このようなとき、偏微分方程式が扱う物理空間を直交格子により分割し、物理量の微分を隣接する格子点との差分で近似し、得られた差分方程式を前処理付き反復法を用いて計算機で解いて、各格子点上での解の近似値を得る方法が使用される。

【０００３】３次元の直方体上での問題の場合は、各辺をＮＸ，ＮＹ，ＮＺ個の格子点で分割したとすると、格子点の集合Ｇは、

Ｇ＝｛（Ｉ，Ｊ，Ｋ）｜１≦Ｉ≦ＮＸ，１≦Ｊ≦ＮＹ，１≦Ｋ≦ＮＺ｝

となり、各格子点において、もとの偏微分方程式を７点差分により近似したときは、各格子点

【０００４】

【数１】

$$(I, K, J) \in G$$

【０００５】において、差分方程式

$$a1(I,J,K)\cdot u(I,J,K-1)+a2(I,J,K)\cdot u(I,J-1,K)+a3(I,J,K)\cdot u(I-1,J,K)+a4(I,J,K)\cdot u(I,J,K)+a5(I,J,K)\cdot u(I+1,J,K)+a6(I,J,K)\cdot u(I,J+1,K)+a7(I,J,K)\cdot u(I,J,K+1)=b(I,J,K)\quad\cdots(1)$$

を得る。ここで、ｕ（Ｉ，Ｊ，Ｋ）が求める未知の物理量に対応し、ａ１・・・ａ７、ｂは偏微分方程式と格子点間の幅より定まるデータである。ただし、ａ１（Ｉ，Ｊ，１）などのＧの外の格子点に掛かる係数は零とする。

【０００６】（１）式をＧ内のすべての格子点に連立させて、一つの連立一次方程式ができる。これを図７のベクトル計算機で解く場合、ａ１・・・ａ７、ｂ，ｕなどの各種ベクトルデータは、計算のベクトル化による高速化のため、通常、１次元配列として主記憶装置上に確保される。例えばベクトルデータｕに対しては、Ｎ＝ＮＸ・ＮＹ・ＮＺとするとき、配列Ｕ（１），・・・，Ｕ（Ｎ）を

$$U(I+(J-1)\cdot NX+(K-1)\cdot NX\cdot NY)=u(I,J,K)\quad\cdots(2)$$

により対応させる。他のデータに関しても同様にする。

ベクトル計算機の主記憶装置は、図７に示すように複数個のメモリバンクＭＢ（０），ＭＢ（１），・・・，Ｍ

B（N－1）により構成される。各メモリバンク性独立に動作できるので、一つのメモリバンクの動作速度が遅くても異なるメモリバンクを連続してアクセスする場合性高速にデータが読み書きできる。メモリ全体のアドレス性図7に示すように連続するワードが異なるメモリバンクに割り当てられるようインターリーブ方式をとる。このとき各配列は、添字が1増加するごとに、一つのデータにつき1ワード要する場合性アドレスを1増加させて、2ワード要する場合性アドレスを2増加させて主記憶装置上に置かれる

【0007】Gに含まれる格子点を順序付け、u（I，J，K）とb（I，J，K）をその順序にしたがってならべた列ベクトル表示をｖｕ，ｖｂとすると、すべての格子点についての（1）式性N×N正方行列Aを用いて、

Ａ・ｖｕ＝ｖｂ・・・（3）

とまとめて表すことができる。この場合Aのほとんどの成分は零であり、このような疎な係数行列をもつ連立一次方程式を解くのに、前処理付き反復解法が有効である。前処理付き反復解法は、内積、Aに対する行列ベクトル積、ベクトルの積和、前処理からなり、これらの組み合わせにより種々の手法が提案されている。したがって、前処理付き反復法を高速に実行するためには、前記の各演算を高速に実行する必要がある。前記の各演算のうち前処理とは、Aの適当な近似行列Pと与えられたベクトルｖｒに対して、Pを係数行列とする連立一次方程式

Ｐ・ｖｓ＝ｖｒ・・・（4）

を解いて、ベクトルｖｓを求める逆近似操作のことである。前処理について注意すべきことは、前処理行列Pが係数行列Aを良く近似しているほど、少ない反復回数で精度の良い近似解が各反復解法において得られることである。このような条件を満たす行列としてAのＭＩＬＵ分解行列がある。ＭＩＬＵ分解行列は、Aの狭義下三角行列L、Aの狭義上三角行列Uと対角行列Dを用いて、

Ｐ＝（Ｌ＋Ｄ）・Ｄ$^{-1}$・（Ｄ＋Ｕ）・・・（5）

と表せる。ここで、Dは0と1の間のパラメータαを一

ｑ（Ｉ，Ｊ，Ｋ）＝ｄ（Ｉ，Ｊ，Ｋ）・（ｒ（Ｉ，Ｊ，Ｋ）－ａ３（Ｉ，Ｊ，Ｋ）・ｑ（Ｉ－１，Ｊ，Ｋ）－ａ２（Ｉ，Ｊ，Ｋ）・ｑ（Ｉ，Ｊ－１，Ｋ）－ａ１（Ｉ，Ｊ，Ｋ）・ｑ（Ｉ，Ｊ，Ｋ－１））・・・（9）

を並列に計算でき、（7）式の後退代入でも、同様にH（NX＋NY＋NZ）からH（3）にかけて順次、各ハ

ｓ（Ｉ，Ｊ，Ｋ）＝ｑ（Ｉ，Ｊ，Ｋ）－ｄ（Ｉ，Ｊ，Ｋ）・（ａ５（Ｉ，Ｊ，Ｋ）・ｓ（Ｉ＋１，Ｊ，Ｋ）＋ａ６（Ｉ，Ｊ，Ｋ）・ｓ（Ｉ，Ｊ＋１，Ｋ）－ａ７（Ｉ，Ｊ，Ｋ）・ｓ（Ｉ，Ｊ，Ｋ＋１））・・・（10）

を並列に計算できる。ここでｓ（Ｉ，Ｊ，Ｋ）、ｑ（Ｉ，Ｊ，Ｋ）、ｒ（Ｉ，Ｊ，Ｋ）は、それぞれ列ベクトルｖｓ，ｖｑ，ｖｒの格子点（Ｉ，Ｊ，Ｋ）に対応する成分の値である。

【0013】マルチカラーオーダリングでは、2以上の

の定めて、

Ｄ＝ｄｉａｇ（Ｌ・Ｄ$^{-1}$・Ｕ）－ｎ・ｒｏｗｓｕｍ（ｎｏｎｄｉａｇ（Ｌ・Ｄ$^{-1}$・Ｕ））＝ｄｉａｇ（Ａ）・・・（6）

より決まる対角行列である。（6）式において、ｄｉａｇ（・）は（）内の行列の対角成分以外を零とする対角行列を、ｎｏｎｄｉａｇ（・）は（）内の行列の対角成分を零とする行列をｒｏｗｓｕｍ（・）は（）内の行列に対し各行の和をその行の対角成分とした対角行列を表す。

【0008】ＭＩＬＵ分解行列に対する前処理は、前進代入と後退代入からなる。前進代入では、

（Ｌ＋Ｄ）・ｖｑ＝ｖｒ・・・（7）

を満たすベクトルｖｑを求め、後退代入では、

Ｄ$^{-1}$・（Ｄ＋Ｕ）・ｖｓ＝ｖｑ・・・（8）

を解いてｖｓを求める。

【0009】A，L，Uの非零成分の位置は列ベクトル表示を定めるときの格子点の順序付けにより異なる。したがって、近似行列Pや（7）式、（8）式を解くときの演算の並列性もこの順序付けにより異なってくる。従来技術では（7）式、（8）式をベクトル計算機で高速に解くために、格子点の順序付けが工夫され、ハイパープレーンオーダリング、マルチカラーオーダリングなどを利用した前進、後退代入の高速化が用いられていた。

【0010】ハイパープレーンオーダリングは図8（a）に示すように格子点をハイパープレーン

【0011】

【数2】

$$H(LH)=\{(I,J,K)\in G \mid I+J+K=LH\}$$
$$(3\leqq LH\leqq NX+NY+NZ)$$

【0012】ごとに、小さなＬＨのハイパープレーンH（ＬＨ）に含まれる格子点から順に並べる順序付けである。このとき、（6）式の前進代入では、H（3）からH（ＮＸ＋ＮＹ＋ＮＺ）にかけて順次、各ハイパープレーンに含まれる格子点ごとに

イパープレーンに含まれる格子点ごとに

カラーの数ＮＣを最初に定め0≦ＬＣ≦ＮＣ－1なる各整数Ｌｃにつきカラー集合Ｃ（ＬＣ）を

【0014】

【数3】

5 6

$$C(LC)=\{(I,J,K)\in G \mid \mathrm{mod}(I+J+K-3,NC)=LC\}$$
$$(0\leqq LC\leqq NC-1)$$

【0015】と定める。そしてLCの小さなC（LC）に含まれる格子点から順序付ける。このとき、各C（LC）は図8（b）に示すようにハイパープレーンの集合となる。（6）式の前進代入はハイパープレーンオーダリングのときと同様にC（0）からC（NC－1）にかけて順次、各カラー集合ごとに並列に、（7）式の後退代入はC（NC－1）からC（0）にかけて順次、各カラー集合ごとに並列に計算できる。この従来のハイパープレーンオーダリングを利用する方法については後保範著、「ベクトル計算機向きICCG法」（京都大学数理解析研究所講究録、No. 514 1984）とJack J. Dongarra他著、「Solving Linear System on Vector and Shared Memory Computer」（SIAM 1991）に、マルチカラーオーダリングを利用する方法についてはJames M. Ortega著、「Introduction to Parallel and Vector Solution of Linear systems」（PLENUM1988）に記述されている。

【0016】

【発明が解決しようとする課題】ところでこのようなオーダリングを利用して、MILU分解行列により前処理を行うとき、並列に計算できる部分はパイプラインプロセッサ上でベクトル化して実行される。この際並列に実行される演算の個数をベクトル長と呼び、パイプライン計算機ではベクトル長が大きいほど稼働効率が増す。また、ベクトル化の際の主記憶の読み書きは規則的であることが好ましい。すなわち、各配列の中のベクトル化の対象となるデータが主記憶上に連続または等間隔に並んでいる場合は、バンク衝突が起こらない限りパイプラインプロセッサと主記憶間のデータ転送が高速に行えるが、不規則な場合はパイプラインプロセッサが1つ1つのデータに関して、その位置を主記憶上に記憶されたリストを参照しながら読み書きを行う必要があり、このオーバーヘッドのためパイプラインプロセッサの稼働効率が落ち演算が遅くなる。

【0017】ハイパープレーンオーダリングを利用する場合、前進後退代入でのベクトル化は各ハイパープレーンごとに行われる。このとき、各配列内の参照されるデータは、式（2）の配列の作り方より一つのyz平面内では、主記憶上でNX－1の間隔で並んでいるが、全体としては不規則である。したがって、ベクトル化に際しては図9のような配列LIST（1），・・・，LIST（NX・NY・NZ），NL（2），・・・，NL（NX＋NY＋NZ）を主記憶上にあらかじめ作成しておく必要がある。ここでLISTには、ハイパープレーンオーダリングの順に各格子点の配列内の位置を格納し、NL（LH）にはH（LH）内の最後の格子点の順序を格納する。ただし、NL（2）＝0とおく。このような、リストを利用すると、前進代入時の計算では、IをNL・（LH－1）＋1からNL（H）まで動かし、J＝LIST（I）として演算

Q（J）＝D（J）・（R（J）－A3（J）・Q（J－1）－A2（J）・Q（J－NX）－A1（J）・Q（J－NX・NY）

を行うことにより、H（LH）上での前進代入ができる。しかし、この場合リストの参照によるオーバーヘッドと、更にLHが3または、NX＋NY＋NZに近いときはH（LH）に含まれる格子点の数が少ないため、ベクトル長が短いという理由から、パイプラインプロセッサの稼働効率が落ちるという問題があった。また、NX－1とバンクメモリの数の最大公約数が大きいときは、H（LH）内の一つのxy平面に含まれる格子点はNX－1飛びで主記憶上に並んでいるため、バンク衝突によりデータの読み書きが遅れ、このため演算速度が落ちるという問題点もあった。

【0018】一方、マルチカラーオーダリングでは、カラーの数NCがNX－1とNY－1の公約数のとき、図10に示すように各C（LC）内の格子点は一つの配列のなかでNC飛びで等間隔に並ぶ。また、ベクトル長も各C（LC）でほぼ等しくなる。このように、前記の条件を満たすNCがある場合、ハイパープレーンオーダリングのように、リストを用いる必要がなく、ベクトル長も一様で長くなるが、NCとバンクメモリの個数の最大公約数が大きいときバンク衝突が起こる。また、この場合C（0）とC（NC－1）の繋ぎ目でのMILU分解行列のAに対する近似度がハイパープレーンオーダリングを用いた場合よりも著しく悪くなるため、ハイパープレーンオーダリングを用いた場合よりも多くの反復を同程度の精度の解を得るまで行う必要が生じるという問題点があった。またNCが小さくなるほどC（0）とC（NC－1）の繋ぎ目の個数が増え収束性が益々悪化するという問題点があった。

【0019】

【課題を解決するための手段】第1の本発明は、パイプラインプロセッサと、複数のメモリバンクよりなりインターリーブ方式を採用した主記憶装置とからなるベクトル計算機システムでの前処理付き反復解法方式において、偏微分方程式のn点差分近似により得られる係数行列の配列データA1，A2，…，An、右辺の配列データB、反復の初期値の配列データU、格子点サイズNX，NY，NZ、多重度KDを入力データとし、前記格子点サイズを参照してXY平面ごとの増分を算出し、各

ＸＹ平面の配列データの末尾に前記ＸＹ平面ごとの増分を追加することにより前記係数行列の配列データ、右辺の配列データ、反復の初期値の配列データを等間隔アクセス形式に並べ換えるデータ並べ換え手段と、前記並べ換え手段で並べ換えられた配列データと多重度ＫＤを参照して、主記憶装置への等間隔アクセスのベクトル演算で前記係数行列の近似行列分解を行い近似行列分解の配列データを生成する多数係数行列分解手段と、前記多重度、前記係数行列分解手段で得た配列データ、前記並べ換え手段で得た係数行列の配列データを参照して、主記憶装置への等間隔アクセスのベクトル演算で前進後退代入を行う多重前進後退代入手段とからなることを特徴とする。

【００２０】また第２の本発明は、パイプラインプロセッサと、複数のメモリバンクよりなりインターリーブ方式を採用した主記憶装置とからなるベクトル計算機システムでの前処理付き反復解法方式において、偏微分方程式のｎ点差分近似により得られる係数行列の配列データＡ１，Ａ２，…，Ａｎ、右辺の配列データＢ、反復の初期値の配列データＵ、格子点サイズＮＸ，ＮＹ，ＮＺ、多重度ＫＤを入力データとし、前記格子点サイズを参照して、前記係数行列の配列データ、右辺の配列データ、反復の初期値の配列データを第１の発明の等間隔アクセス形式への並べ換え手段により等間隔アクセス形式に並べ換えたのち、等間隔にならんだデータを連続に並べ換える連続アクセス形式へのデータ並べ換え手段と、前記並べ換え手段で並べ換えられた配列データと多重度ＫＤを参照して、主記憶装置への連続アクセスのベクトル演算で前記係数行列の近似行列分解を行い近似行列分解の配列データを生成する多重係数行列分解手段と、前記多重度、前記係数行列分解手段で得た配列データ、前記並べ換え手段で得た係数行列の配列データを参照して、主記憶装置への連続アクセスのベクトル演算で前進後退代入を行う多重前進後退代入手段とからなることを特徴とする。

【００２１】

【作用】３次元７点差分近似を例にして説明する。本発明では、カラーの数ＮＣをＮＸ－１として、格子点全体の集合Ｇをカラー集合Ｃ（０），・・・，Ｃ（ＮＸ－２）に分けて、係数行列分解、前進後退代入はこのカラー集合ごとにベクトル化して行う。ベクトル化の際、各ベクトル変数に対応する配列内で一つのカラー集合のデータがＮＸ－１の等間隔、または連続に並ぶように次のようなデータの並べ換え方式を用いる。

【００２２】等間隔アクセス形式への並べ換え方式では、各ベクトル変数に対応する配列サイズをＭ＝ＮＸ・（ＮＹ＋１）・ＮＺだけ確保し、入力ベクトル変数のＡ１，・・・，Ａ７，Ｂに関しては各配列の１からＮＸ・ＮＹ・ＮＺまでの添字に対して必要とするデータを格納しておく。最初に、

ｍｏｄ（ＮＸ・ＮＹ＋ＩＮＣ－１，ＮＸ－１）＝０，０≦ＩＮＣ≦ＮＸ－１

を満たす非負整数ＩＮＣを

ＩＮＣ０＝ＮＸ－１－ｍｏｄ（ＮＹ－１，ＮＸ－１），

ＩＮＣ＝ｍｏｄ（ＩＮＣ０，ＮＸ－１）・・・（１１）

より計算する。例えば、配列Ｖの並べ換えにおいては、図２に示すように作業用配列Ｗ（１），・・・，Ｗ（Ｍ）をメモリ内に確保し、最初にＷのすべての要素に指定した値Ｃを書き込んでおく。次にＷにＶの値をｘｙ平面ごとに間隔ＩＮＣをおきながら写し、最後にＷの値をＶに写す。これにより図１（ａ）に示すように１つのカラー集合内のデータがＮＸ－１の等間隔で並ぶようになり、各ｘｙ平面ごとにＩＮＣだけ追加した部分には定数Ｃが残る。

【００２３】連続アクセス形式への並べ換え方式では、各カラー集合内の格子点の数の上限に近い上界ＮＵを

ＮＵ＝ｉｎｔ（（Ｍ－ＮＸ）／（ＮＸ－１））・・・（１２）

より計算する。ここで、ＩＮＴ（・）は（）内の実数の整数部のみをとることを意味する。並べ換えでは、図２に示すように上記等間隔アクセス形式の場合と同様にして配列Ｗに並べ換えたい配列Ｖを書き込む。次に、ＬＣの小さいカラー集合Ｃ（ＬＣ）からそれに対応するＷの要素をＶに連続に書き込む。これにより図１（ｂ）に示すように１つのカラー集合内のデータが連続に並ぶようになる。

【００２４】多重係数行列分解では、各カラー集合ごとに対角行列ｄ１，ｄ２の成分を求める。図３に示すように、多重度ＫＤ（１≦ＫＤ≦ＮＸ－２）を参照して、Ｃ（ＮＸ－１－ＫＤ）上のｄ１の値を１／ａ４と置き、次にＣ（ＮＸ－ＫＤ）からＣ（ＮＸ－２）までの格子点上のｄ１の値を３１０に示すように、ハイパープレーンオーダリングでのＭＩＬＵ分解と同じ式

$$FD(I,J,K,d)=a4(I,J,K)-a3(I,J,K)\cdot d(I-1,J,K)\cdot(a5(I-1,J,K)+\alpha\cdot(a6(I-1,J,K)+a7(I-1,J,K)))-a2(I,J,K)\cdot d(I,J-1,K)\cdot(a6(I,J-1,K)+\alpha\cdot(a5(I,J-1,K)+a7(I,J-1,K)))-a1(I,J,K)\cdot d(I,J,K-1)\cdot(a7(I,J,K-1)+\alpha\cdot(a5(I,J,K-1)+a6(I,J,K-1)))\cdots(13)$$

でカラー集合ごとに順次求める。この際Ｃ（ＬＣ）上の格子点（Ｉ，Ｊ，Ｋ）での計算において、必要とされる

右辺のｄ１は格子点（Ｉ－１，Ｊ，Ｋ），（Ｉ，Ｊ－１，Ｋ），（Ｉ，Ｊ，Ｋ－１）上の値であり、これらはすべてＣ（ＬＣ－１）に含まれるから、一つのカラー集合上での計算は並列に行われる。ｄ１の他の値については、後で使用しないので求めずともよい。次にｄ２の計算を全格子点について順次Ｃ（０）からＣ（ＮＸ－２）まで行う。ただし、図３の３２０のＣ（０）での計算では、右辺でＣ（ＮＸ－２）上のｄ１の値を参照する。

$$FS(I,J,K,d,r,q)=d(I,J,K)\cdot(r(I,J,K)-a3(I,J,K)\cdot q(I-1,J,K)-a2(I,J,K)\cdot(I,J-1,K)-a1(I,J,K)\cdot q(I,J,K-1))\cdots(14)$$

Ｃ（０）での計算の前にＣ（ＮＸ－１－ＫＤ）からＣ（ＮＸ－２）まで前進代入を行いＣ（０）での計算では、先の前進代入で求めたＣ（ＮＸ－２）上での値を参照することにより、多重度ＫＤが大きいほどここで求めたｑはハイパープレーンオーダリングでの前進代入の結果に近づく。また各Ｃ（ＬＣ）上の格子点（Ｉ，Ｊ，Ｋ）での計算において、必要とされる右辺のｑは格子点（Ｉ－１，Ｊ，Ｋ），（Ｉ，Ｊ－１，Ｋ），（Ｉ，Ｊ，Ｋ－１）上の値であり、これらはすべてＣ（ＬＣ－１）に含まれるから、一つのカラー集合上での計算は並列に行われる。

【００２５】次に多重前進後退代入方式の説明を行う。図４に示すように前進代入では、４１０でＣ（ＮＸ－１－ＫＤ）からＣ（ＮＸ－２）まで、順次ｄ１を参照してｑの値を求める。次に４２０でｄ２を参照してＣ（０）からＣ（ＮＸ－２）までのｓの値を求める。ここで各格子点における演算式ＦＳ（・）はハイパープレーンオーダリングでの演算式と同じものであり、次式に従う。

$$BS(I,J,K,d,q,s)=q(I,J,K)-d(I,J,K)\cdot(a5(I,J,K)\cdot s(I+1,J,K)+a6(I,J,K)\cdot s(I,J+1,K)-a7(I,J,K)\cdot s(I,J,K+1))\cdots(15)$$

次に４４０で、Ｃ（０）でのｓの値を参照してＣ（ＮＸ－２）でのｔの値を求め、ｔの値をｓに加える。次にＣ（ＮＸ－３）からＣ（ＮＸ－１－ＫＤ）まで順次、前のステップで求めたｔを参照しながらｔを求めｓに加えていく。この後退代入においても各Ｃ（ＬＣ）上の格子点（Ｉ，Ｊ，Ｋ）での計算において、必要とされる右辺のｓまたはｔは格子点（Ｉ＋１，Ｊ，Ｋ），（Ｉ，Ｊ＋１，Ｋ），（Ｉ，Ｊ，Ｋ＋１）上の値であり、これらはすべてＣ（ＬＣ＋１）に含まれるから、一つのカラー集合上での計算は並列に行われる。

【００２６】後退代入では、上記前進代入と対象な操作を行う。４３０でＣ（ＮＸ－２）からＣ（０）にかけて順次後退代入を行いｓを求める。このとき演算式はハイパープレーンオーダリングと同じものを用い次式に従う。

【００２７】上記のように、Ｃ（ＮＸ－１－ＫＤ）からＣ（ＮＸ－２）での前進後退代入を２回行うことにより、適当なＫＤを選べばハイパープレーンオーダリングでのＭＩＬＵ分解行列による前処理と同様の収束性をもつ前処理が実現できる。なお上記の方法は一般の差分近似に拡張できる。

【００２８】

【実施例】次に本発明の実施例について３次元７点差分を例にして、図面を参照して詳細に説明する。

【００２９】図５は本発明により、等間隔アクセス形式で前処理付き共役勾配法を実行するときの処理手順であり、図４のＤＯ　ＰＡＲＡＬＬＥＬで示す各ベクトル演算がＮＸ－１飛びの等間隔アクセスで実行できる。例えば、４２０のＣ（ＬＣ）上でのベクトル演算は、

$$Y(J)=A1(J)\cdot V(J-NXYC)+A2(J)\cdot V(J-NX)+A3(J)\cdot V(J-1)+A4(J)\cdot V(J)+A5(J)\cdot V(J+1)+A6(J)\cdot V(J+NX)+A7(J)\cdot V(J+NXYC)$$

を添字Ｊにつき１からＭ＝ＮＸ・（ＮＹ＋１）・ＮＺまで行えばよい。

【００３１】前処理も、一つの格子点の配列データの添字とその周りの格子点の配列データの添字の規則性よる。５１０の配列の並べ換えでは、係数行列の対角成分にあたるＡ４のみは、図２の２１０の定数Ｃとして１を用い、他のＡ１，Ａ２，Ａ３，Ａ５，Ａ６，Ａ７，Ｂ，初期値Ｕについては０を用いて並べ換える。これは、５２０で行う係数行列分解で零割りが生じるのを防ぐためである。５３０で初期設定を行う。反復計算では、５４０と５７０で積和、５６０と５９２で内積、５５０で行列ベクトル積、５９０で前処理を行う。５８０では収束判定を行い残差ｒが十分小さければ、５９１で、Ａ１，・・・，Ａ７，Ｂと得られた解Ｕをもとのデータ形式に並べ換え終了する。

【００３０】内積、積和計算はＲ（ＮＸ・ＮＹ＋１），・・・，Ｒ（ＮＸ・ＮＹ＋ＩＮＣ）などの並べ換えのとき追加したデータもこめて行えば１度の連続アクセスのベクトル演算でできる。ただし、どのベクトル変数についても、追加分には反復の初期に０を入れておく。行列ベクトル積も隣合う格子点の添字の関係が図６（ａ）に示すように規則的になるから、５３０と５５０での行列ベクトル積も次のように１回の連続アクセスのベクトル演算で済む。例えば、５５０では

【００３１】前処理も、一つの格子点の配列データの添字とその周りの格子点の配列データの添字の規則性よ

Q（J）＝D2（J）・（R（J）－A3（J）・Q（J－1）－A2（J）・Q（J－NX）－A1（J）・Q（J－NXYC））

をJ＝LC＋1からMまでNX－1個ずつで行えばよい。図3に示した係数行列分解でD1，D2を求めるときも同様である。

【0032】第2の実施例は、連続アクセス形式のデータを使用する場合である。この場合も、上記第1の実施例と同様で図5に示す処理の流れに従う。ただし、図6

Y（J）＝A1（J）・V（J＋NUZ－IXY－1）＋A2（J）・V（J－NUZ－2）＋A3（J）・V（J＋NUZ－1）＋A4（J）・V（J）＋A5（J）・V（J＋NU）＋A6（J）・V（J＋NU＋1）＋A7（J）・V（J＋NU1）

をJ＝1からNUまで連続に1回のベクトル演算で行い、次に

Y（J）＝A1（J）・V（J－NU1）＋A2（J）・V（J－NU－1）＋A3（J）・V（J－NU）＋A4（J）・V（J）＋A5（J）・V（J＋NU）＋A6（J）・V（J＋NU＋1）＋A7（J）・V（J＋NU1）

をJ＝NU＋1からNU・（NX－2）まで連続に1回のベクトル演算で行い、

最後に

Y（J）＝A1（J）・V（J－NU1）＋A2（J）・V（J－NU－1）＋A3（J）・V（J－NU）＋A4（J）・V（J）＋A5（J）・V（J－NUZ＋1）＋A6（J）・V（J－NUZ＋2）＋A7（J）・V（J－NUZ＋IXY＋1）

をJ＝NU・（NX－2）＋1からNU・（NX－1）まで連続に1回のベクトル演算で行えばよい。前処理に関しては、図4のDO　PARALLELで示す各ベクトル演算が連続アクセスで実行できる。例えば、420のC（LC）上でのベクトル演算は、

Q（J）＝D2（J）・（R（J）－A3（J）・Q（J－NU）－A2（J）・Q（J－NU－1）－A1（J）・Q（J－NU1））

をJ＝NU・LC＋1からNU・（LC＋1）まで連続して行えばよい。図3に示した係数行列分解でD1，D2を求めるときも同様である。

【0033】

【発明の効果】以上説明したように本発明により、前進後退代入でのベクトル演算のベクトル長がマルチカラーオーダリングと同様にカラー集合内の格子点の数だけとれるため十分長くなり、多重度で指定された分だけ重ねて前進後退代入を行うことにより、マルチカラーオーダリングのように収束性が悪化せず、ハイパープレーンオーダリングと同様の収束性を示す前処理が実現できる。また、配列の等間隔アクセス形式への並べ換えにより、任意の格子点サイズNX，XY，NZの問題に対し、一つのカラー集合内の配列データがNX－1の等間隔でアクセスでき、ハイパープレーンオーダリングでの前進後退代入に比べてパイプラインプロセッサの稼働効率が良くなる。さらに、連続アクセス形式への並べ換えを使用すれば、一つのカラー集合内の配列データがアクセスでき、どのような格子点サイズの問題に対してもバンク衝突でパイプラインプロセッサの稼働効率が落ちることがなくなる。

【図面の簡単な説明】

【図1】（a）は等間隔アクセス形式の配列並びを、（b）に示すように、各格子点とそのまわりの格子点の添字の関係が、その格子点がどのカラー集合に含まれるかにより異なる。したがって、行列ベクトル積では、3つに分けてベクトル演算をする必要がある。例えば、50では

Y（J）＝A1（J）・V（J＋NUZ－IXY－1）＋A2（J）・V（J－NUZ－2）＋A3（J）・V（J＋NUZ－1）＋A4（J）・V（J）＋A5（J）・V（J＋NU）＋A6（J）・V（J＋NU＋1）＋A7（J）・V（J＋NU1）

をJ＝1からNUまで連続に1回のベクトル演算で行い、次に

Y（J）＝A1（J）・V（J－NU1）＋A2（J）・V（J－NU－1）＋A3（J）・V（J－NU）＋A4（J）・V（J）＋A5（J）・V（J＋NU）＋A6（J）・V（J＋NU＋1）＋A7（J）・V（J＋NU1）

をJ＝NU＋1からNU・（NX－2）まで連続に1回のベクトル演算で行い、

最後に

Y（J）＝A1（J）・V（J－NU1）＋A2（J）・V（J－NU－1）＋A3（J）・V（J－NU）＋A4（J）・V（J）＋A5（J）・V（J－NUZ＋1）＋A6（J）・V（J－NUZ＋2）＋A7（J）・V（J－NUZ＋IXY＋1）

をJ＝NU・（NX－2）＋1からNU・（NX－1）まで連続に1回のベクトル演算で行えばよい。前処理に関しては、図4のDO　PARALLELで示す各ベクトル演算が連続アクセスで実行できる。例えば、420のC（LC）上でのベクトル演算は、

Q（J）＝D2（J）・（R（J）－A3（J）・Q（J－NU）－A2（J）・Q（J－NU－1）－A1（J）・Q（J－NU1））

をJ＝NU・LC＋1からNU・（LC＋1）まで連続して行えばよい。図3に示した係数行列分解でD1，D2を求めるときも同様である。

【0033】

【発明の効果】以上説明したように本発明により、前進後退代入でのベクトル演算のベクトル長がマルチカラーオーダリングと同様にカラー集合内の格子点の数だけとれるため十分長くなり、多重度で指定された分だけ重ねて前進後退代入を行うことにより、マルチカラーオーダリングのように収束性が悪化せず、ハイパープレーンオーダリングと同様の収束性を示す前処理が実現できる。また、配列の等間隔アクセス形式への並べ換えにより、任意の格子点サイズNX，XY，NZの問題に対し、一つのカラー集合内の配列データがNX－1の等間隔でアクセスでき、ハイパープレーンオーダリングでの前進後退代入に比べてパイプラインプロセッサの稼働効率が良くなる。さらに、連続アクセス形式への並べ換えを使用すれば、一つのカラー集合内の配列データがアクセスでき、どのような格子点サイズの問題に対してもバンク衝突でパイプラインプロセッサの稼働効率が落ちることがなくなる。

【図面の簡単な説明】

【図1】（a）は等間隔アクセス形式の配列並びを、（b）は連続アクセス形式の配列並びを示す図である。

【図2】配列の等間隔アクセス形式または連続アクセス形式への並べ換え手段を示す図である。

【図3】多重度KDの多重係数行列分解手段を示す図である。

【図4】多重度KDの多重前進後退代入手段を示す図である。

【図5】前処理付き共役勾配法を示す図である。

【図6】（a）は等間隔アクセス形式における添字の関係を、（b）は連続アクセス形式における添字の関係を示す図である。

【図7】ベクトル計算機システムを示す図である。

【図8】（a）はハイパープレーンオーダリングでの順序付けを、（b）はマルチカラーオーダリングでの順序付けを示す図である。

【図9】ハイパープレーンオーダリングのリストの構造を示す図である。

【図10】カラー集合と配列の添字の対応を示す図である。

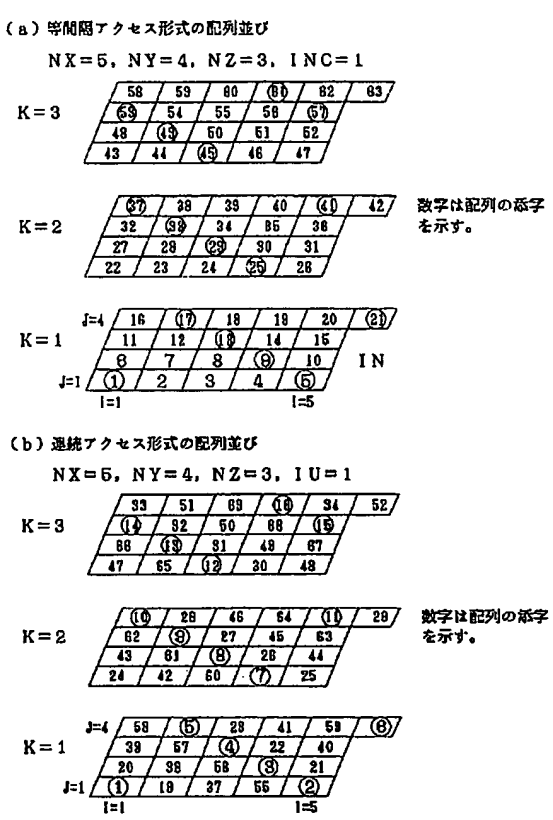
【符号の説明】

710　パイプラインプロセッサ

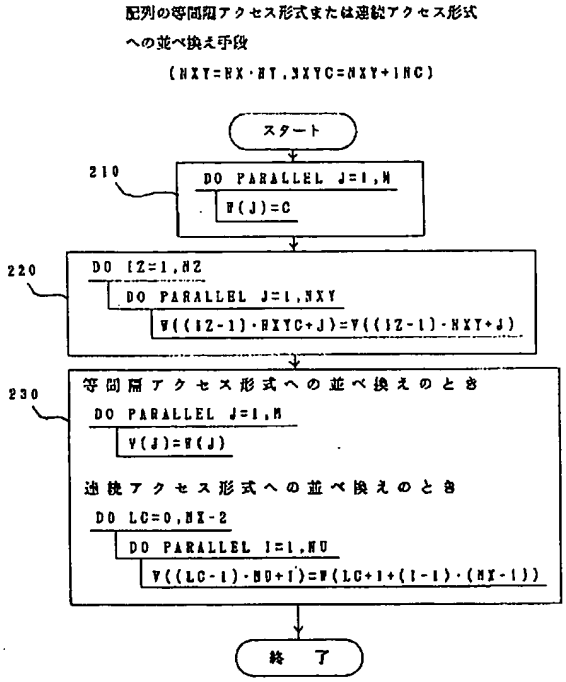
720　主記憶装置

730　メモリバンク

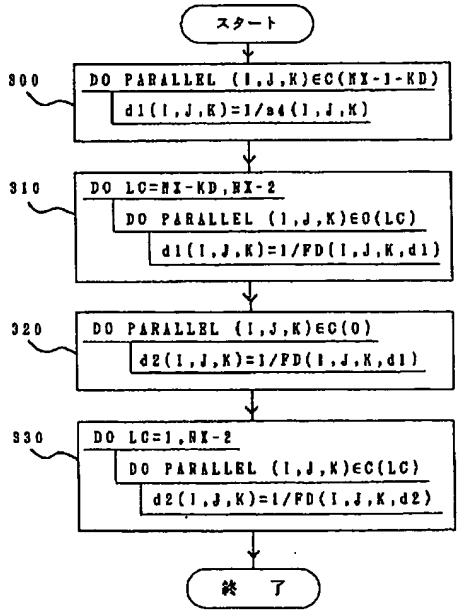
【図1】


(a) 等間隔アクセス形式の配列並び
NX=5, NY=4, NZ=3, INC=1
K=3
K=2
K=1
J=4
J=1
I=1
I=5
IN
数字は配列の添字を示す。
(b) 連続アクセス形式の配列並び
NX=5, NY=4, NZ=3, IU=1
K=3
K=2
K=1
J=4
J=1
I=1
I=5
数字は配列の添字を示す。

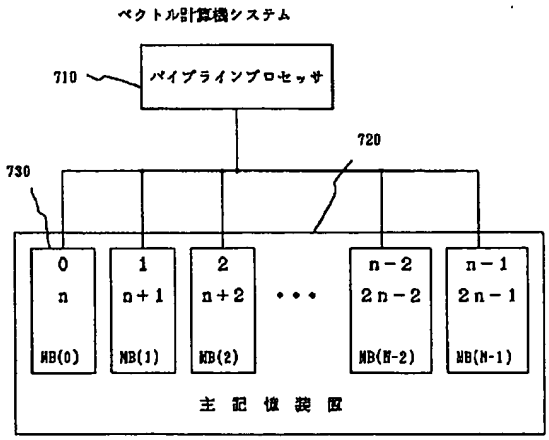
【図2】


配列の等間隔アクセス形式または連続アクセス形式への並べ換え手段
(NXY=NX・NY, NXYC=NXY+INC)
スタート
210
DO PARALLEL J=1,N
W(J)=C
220
DO IZ=1,NZ
DO PARALLEL J=1,NXY
W((IZ-1)・NXYC+J)=W((IZ-1)・NXY+J)
230
等間隔アクセス形式への並べ換えのとき
DO PARALLEL J=1,N
V(J)=W(J)
連続アクセス形式への並べ換えのとき
DO LC=0,NX-2
DO PARALLEL I=1,NU
V((LC-1)・NU+I)=W(LC+1+(I-1)・(NX-1))
終　了

【図3】

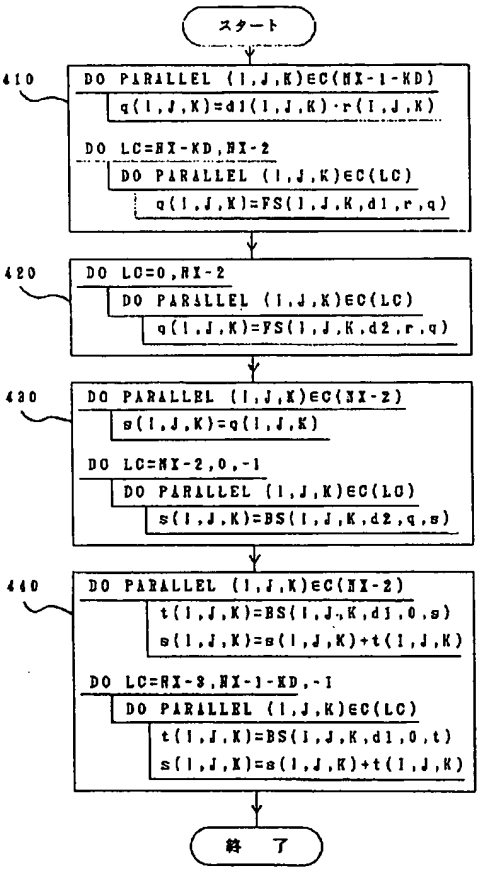

多重度KDの多重係数行列分解手段
スタート
300
DO PARALLEL (I,J,K)∈C(NX-1-KD)
d1(I,J,K)=1/a4(I,J,K)
310
DO LC=NX-KD,NX-2
DO PARALLEL (I,J,K)∈C(LC)
d1(I,J,K)=1/FD(I,J,K,d1)
320
DO PARALLEL (I,J,K)∈C(0)
d2(I,J,K)=1/FD(I,J,K,d1)
330
DO LC=1,NX-2
DO PARALLEL (I,J,K)∈C(LC)
d2(I,J,K)=1/FD(I,J,K,d2)
終　了

【図7】


ベクトル計算機システム
710
パイプラインプロセッサ
720
730
0
n
MB(0)
1
n+1
MB(1)
2
n+2
MB(2)
n-2
2n-2
MB(N-2)
n-1
2n-1
MB(N-1)
主記憶装置

【図４】

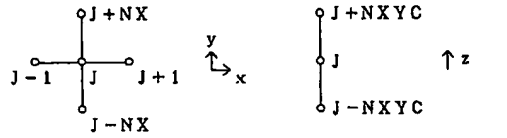
多重度ＫＤの多重前進後退代入手段



【図６】

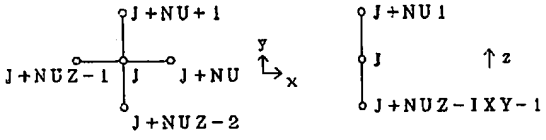
(a) 等間隔アクセス形式における添字の関係

NXYC=NX･NY+1NC

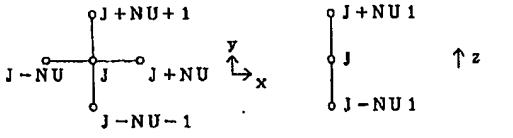


(b) 連続アクセス形式における添字の関係

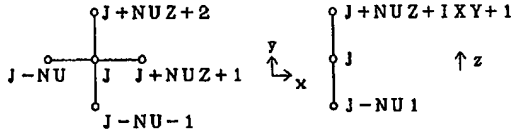
J∈C (0) のとき



J∈C (1) ～C (NX-3) のとき



J∈C (NX-2) のとき



ただし　1XY=(NXYC-1)／(NX-1)

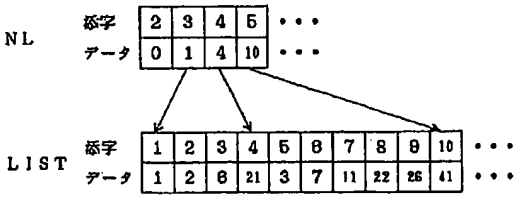
NU1=NU+1XY

NUZ=NU･(NX-2)

【図９】

ハイパープレーンオーダリングのリストの構造

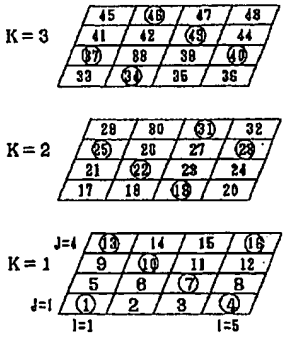
NX=5, NY=4, NZ=3のとき



【図１０】

カラー集合と配列の添字の対応

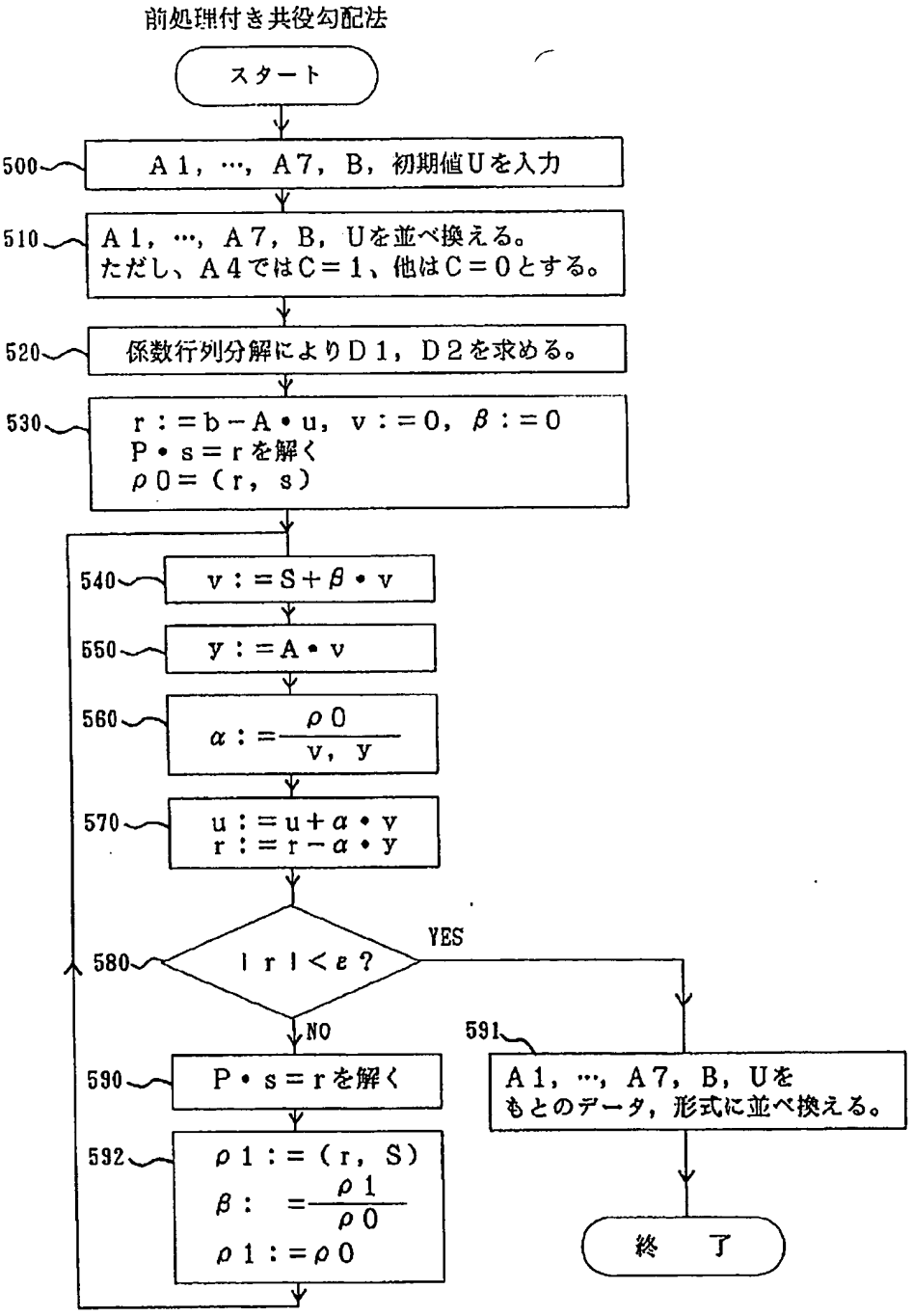
NX=4, NY=4, NZ=3, NC=3



○はC (0) の格子点

数字は配列の添字を示す。

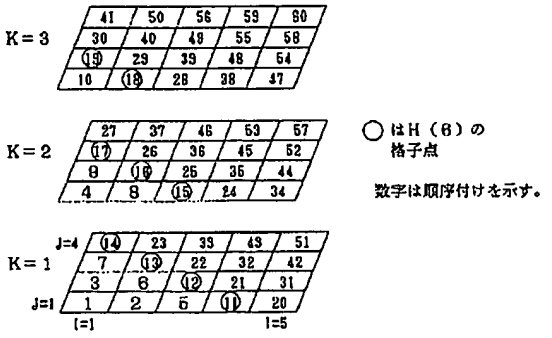
【図５】


前処理付き共役勾配法
スタート
500
A１，…，A７，B，初期値Uを入力
510
A１，…，A７，B，Uを並べ換える。
ただし、A４ではC＝１、他はC＝０とする。
520
係数行列分解によりD１，D２を求める。
530
r：＝b－A・u，v：＝０，β：＝０
P・s＝rを解く
ρ０＝（r，s）
540
v：＝S＋β・v
550
y：＝A・v
560
α：＝ρ０／（v，y）
570
u：＝u＋α・v
r：＝r－α・y
580
|r|＜ε？
YES
NO
590
P・s＝rを解く
592
ρ１：＝（r，S）
β：＝ρ１／ρ０
ρ１：＝ρ０
591
A１，…，A７，B，Uを
もとのデータ，形式に並べ換える。
終　了

【図８】

（ａ）ハイパープレーンオーダリングでの順序付け

NX＝５，NY＝４，NZ＝３

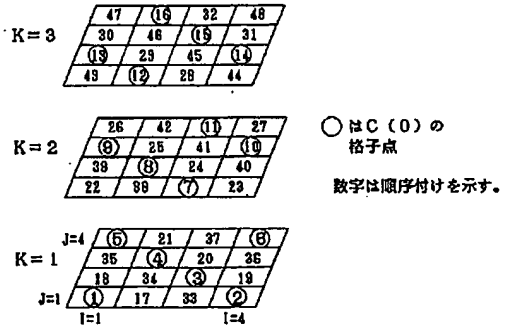


○はＨ（６）の格子点

数字は順序付けを示す。

（ｂ）マルチカラーオーダリングでの順序付け

NX＝４，NY＝４，NZ＝３，NC＝３



○はＣ（０）の格子点

数字は順序付けを示す。